# Integrated Tracker for STAR

*Andrew Rose, Wayne State University, for the STAR Collaboration*

## Goal

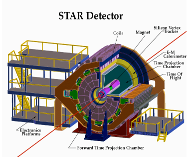



Figure 1: The STAR Detector.

With the introduction of new tracking detectors designed to complement the Time Projection Chamber (TPC), the STAR Collaboration committed to the development of a new, integrated tracking software package to replace the existing disparate packages, unique to each detector system, and maintained individually.

### Requirements for the new tracker:

- Physics Performance

  High efficiency (99% for all tracks at low multiplicity, 90% for analysis like cuts at low multiplicity)

  Good Resolution
- Speed
- Stability

  Previous tracking code was unstable and limited reconstruction to several hundred events per instance. Current code has smaller, stable memory footprint – allowing more flexible use of computing resources.
- Easily Maintained and Upgraded

  Object Oriented design allows for easy package maintenance and upgrades

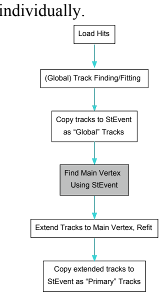



Figure 3: Tracker responsibilities in the STAR reconstruction chain.

## Design

### Code Structure

Component Abstraction – Object Oriented/Abstract Interfaces

- Not specific to any experiment.
- Simple Geometry Model (Detector, Shape, Placement).
- Elementary Constructs (Track, Hit, etc).
- Special Containers (Detector Geometry, Hits)
- Generic Algorithm/Interface

  Track Seed finder, Track finder, MCS/Eloss/Dedx Calculations
- Templated Object Factory and Memory Management.
- Abstract Input/Control Parameter Representation.

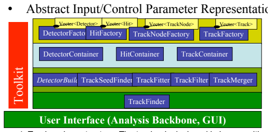



Figure 4: Tracker class structure. The tracker is designed in layers, with a toolkit providing communication of key data and helper classes across the structure. The layers isolate memory structures (factories and containers) from the user specific code.

### Memory Management: Factories

- Use STL vector class for storage.
- Memory allocation and garbage collection done in one place
- Nominal set of object instantiated once at startup and destroyed on exit.
- Object set expanded in large blocks as needed.

#### Advantages

- Avoid repeated calls to "new" and "delete" for each event analysis.
- Enables plug and play of new components.
- Enables run time choice of classes to instantiate and use.
- Simplified user code – no memory management.
- No memory leaks for "factorized" classes.

### Track Representation

- Helix defined in local coordinates of the detector
- Representation rotation required only while switching "sector".
- Use Kalman Filter/Fit
  - State Vector: y, z, η, C, tan(λ)
  - Prediction is simple and fast
  - Account for multiple coulomb scattering and energy loss using knowledge of the detector materials and trajectory.
  - Update of the fit only if a hit is found.

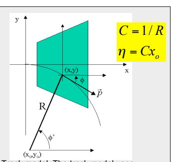



Figure 5: Track model. The track model uses coordinates local to the detector, allowing for faster hit searching.

### Track Finding and Fitting

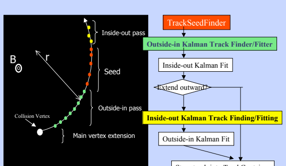



Figure 6: Tracking algorithm.

The tracking algorithm has three major steps: seed finding, outside-in pass, and inside-out pass. A track seed is found by locating hit pairs in R, phi space, then using a simple circle fit to project inward. The track seed is typically 5 hits.

Seeds are then extended inward, toward the origin, using Kalman Fitting to update the track kinematics and project the updated track to the next layer.

When the projection inward is complete – either because the innermost layer is reached, or many successive layers do not have hits which can be added – the tracker steps from the innermost layer out, updating each track node. When the outermost node is reached, the tracker will attempt to extend the track outward beyond the seed as far as possible. If new hits are added in this outward projection, another inward fitting cycle is initiated.

When the fitting is complete, the track is stored and used by the STAR vertex finder. If a vertex is found, the tracker tests each track to see if the vertex is a viable point to include in the track fit; if so, the vertex is added, the track is refit and labeled as a primary.

## Performance

### Track Reconstruction - Hits

Hit residuals on the order of 1mm in the TPC provide high accuracy track definition.

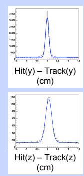


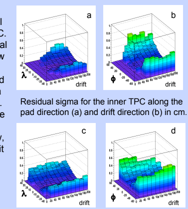

Figure 7: Track residuals in the TPC. Angle (λ,φ)<10, 50<drift<75

Figure 8:

Distributions of residual sigma for the inner TPC. The width of the residual sigma along the padrow is a function of the drift and crossing angle, and in the drift direction is a function of drift and dip. The residual sigmas are fit using a parameterization below, and the results of this fit are used as the expected error in the Kalman fit.

$$\Delta y = \alpha_{y,0} + \alpha_{y,drift}\frac{z_{drift}}{\cos^2\theta} + \alpha_{y,cross}\tan^2\theta$$

$$\Delta z = \alpha_{z,0} + \alpha_{z,drift}\frac{z_{drift}}{\cos^2\lambda} + \alpha_{z,dip}\tan^2\lambda$$




Residual sigma for the inner TPC along the pad direction (a) and drift direction (b) in cm.

Residual sigma for the outer TPC along the pad direction (c) and drift direction (d) in cm.

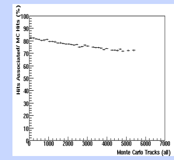



Figure 9:

Monte Carlo hits associated with the proper track as a function of track multiplicity. Most inefficiency at low multiplicity is due to losses at sector boundaries, with higher loss as the hit density increases.

### Track Kinematics Reconstruction: Bias and Resolution

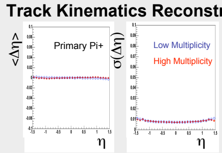



Figure 10: η bias and resolution. The new tracker has a bias, or mean difference between Monte Carlo and reconstructed η, near zero. The resolution, sigma of the difference distribution, is .01 units in the range of interest.

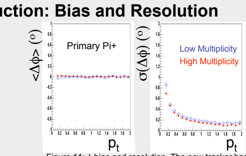



Figure 11: φ bias and resolution. The new tracker has a φ bias near zero. The φ resolution depends strongly on the track pt; in the low pt region, multiple scattering dominates the φ resolution. At higher $p_t$, the best φ resolution is achieved as .2 degrees.

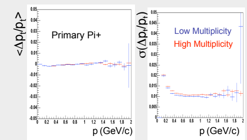



Figure 12: $p_t$ fractional bias and resolution. The new tracker has a pt bias less than .2%. The pt resolution reaches the best value of 1% at a momentum of 650 MeV/c.

### Reconstruction Pulls

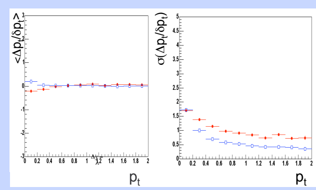



Figure13: Mean and sigma of the transverse momentum pull. Here again, the absence of a bias in the reconstruction of the track's pt is evident. The width is significantly different from the ideal (1) at low momentum where the multiple scattering dominates, but quickly converges at higher momentum.

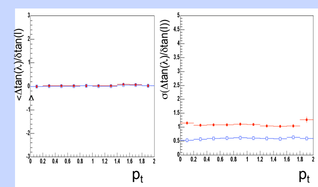



Figure 14: Mean and sigma of the transverse momentum pull. Here again, the absence of a bias in the reconstruction of the track's pt is evident. The width is significantly different from the ideal (1) at low momentum where the multiple scattering dominates, but quickly converges at higher momentum.

### Reconstruction Efficiencies

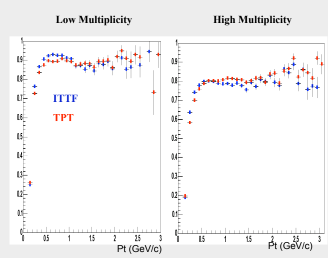



Figure 14: Efficiency as a function of pt. The efficiency here is defined as the number of reconstructed tracks matched to a monte carlo track divided by the number of monte carlo tracks, within a particular kinematic bin. Additional cuts include: -1< η < 1, pt > 150MeV, MC Hit > 20. Low and high multiplicity events shown. Red points denote performance of previous tracking software.

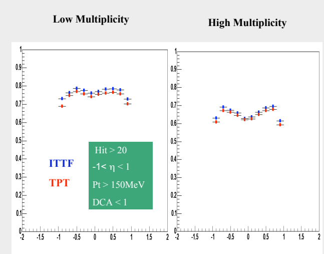



Figure 15: Efficiency as a function of η. Additional cuts include: -1< η < 1, pt > 150MeV, MC Hit > 20. Red points denote performance of previous tracking software.

## Summary

**The new STAR track reconstruction software meets all the identified design goals:**

- **Easy maintenance and expansion through Object Oriented design**
- **Fast, robust algorithms and Factorized memory management provide software stability**
- **Kalman Filtering and Fitting provide accurate track identification and reconstruction**
- **Estimations of multiple scattering and energy loss during the tracking phase allow corrections for these effects.**
- **Performance exceeds expectations**